J. Symbolic Computation (1991) **12**, 391–395

Introduction

Group theory has proved to be a particularly fertile field for the development of effective computational techniques. A variety of useful algorithms have been designed for computing detailed information concerning the structure, representations and extensions of various types of finite group. Techniques have also been developed for studying finitely presented infinite groups.

This special issue is the second in a two-part series devoted to Computational Group Theory. The first appeared as Volume 9, Numbers 5 & 6 (May/June 1990), and was concerned with the areas of representation theory and soluble groups. The current issue covers finitely presented groups (*fp-groups*) and permutation groups.

A well-known theorem, proven independently by Novikov and Boone, asserts that the word problem for fp-groups is, in general, unsolvable. Thus, the design of computational procedures for studying fp-groups is particularly challenging. Let G and K be two fp-groups. Typical of the kinds of questions mathematicians wish to answer about fp-groups are the following:

Is G the trivial group?
Is G finite?
If G is infinite, is it free?
If G is finite, what is its order and structure?
What are the abelian (nilpotent, soluble, perfect) quotients of G?
Is G abelian (nilpotent, soluble, perfect)?
Can we construct a small degree permutation representation for G?
Can we construct a small degree matrix representation for G over some given field?
Are the groups G and K isomorphic?

Two very general techniques available for studying fp-groups are the Todd–Coxeter and the Knuth–Bendix procedures. Given an fp-group G and a subgroup H of G, the classical Todd–Coxeter procedure (*TC-procedure*) attempts to construct a permutation representation for G, corresponding to the action of G on the coset space of H, by means of a trial-and-error process. This procedure was used extensively in hand computation prior to the development of computers. Beginning in 1952, various versions of the procedure have been adapted for machine computation and the TC-procedure is perhaps the most widely applied technique in computational group theory. However, despite its antiquity, our understanding of the relationship between a given presentation for G and the performance of a particular version of the TC-procedure when applied to that presentation is extremely poor. Recently, after extensive experimentation, Havas devised a new Todd–Coxeter strategy, which exhibits dramatically better performance than the traditional versions when applied to many "difficult" enumerations. Unfortunately, this work was not completed in time for inclusion in this volume.

The range of applicability of current Todd–Coxeter programs is limited mainly by the memory required to store a table giving the action of G on the cosets of H (the *coset table*). During a difficult coset enumeration, the same coset may have many different

0747-7171/91/100391+05 $03.00/0

definitions so that the space required to store intermediate tables is often a great deal larger than that required to store the final coset table. Since, however, for non-pathological enumerations, the space required is roughly proportional to the index of H in G, workers in the field have long dreamt of generalizing the TC-procedure to a method capable of enumerating the double cosets HxL of subgroups H and L of G. A significant step towards this goal is made in Linton's paper "Double coset enumeration" which describes a successful implementation of a double coset enumeration procedure for the case where H is a "large" subgroup of G, and L is a small subgroup in which detailed structural computation is possible.

As noted at the outset, the classical Todd-Coxeter procedure constructs a permutation representation for G on the cosets of H. In a second paper, "Constructing matrix representations of finitely presented groups", Linton describes a version of the TC-procedure which constructs a matrix representation for G over a designated field k (usually a finite field). In the simplest interpretation, Linton's algorithm constructs the permutation module corresponding to the action of G on the cosets of a subgroup H. However, since the algorithm works by constructing representations of the group algebra kG, given a suitable choice of ideal generators in kG, it is possible, for example, to construct directly constituents of a permutation module for G.

Starting with a finitely presented monoid M, the Knuth-Bendix procedure (*KB-procedure*) attempts to construct a *confluent presentation* for M. A confluent presentation P for M has the property that there exists a unique normal form for the elements of M relative to P, and the KB-procedure may be used to compute the normal form for any word w of M. Although the KB-procedure has been studied extensively since its publication in 1970, until recently it has rarely proved superior to the TC-procedure when used to solve word problems for fp-groups.

A major advantage of the KB-procedure over the TC-procedure is that it can sometimes construct confluent presentations for infinite groups. In the case of a finite fp-group, the TC-procedure is usually the most efficient method for constructing a confluent presentation. The paper entitled "The use of Knuth-Bendix methods to solve the word problem in automatic groups" by Epstein *et al.* describes a practical algorithm based on the KB-procedure for constructing a solution to the word problem for the class of groups known as *automatic groups*. This class of infinite groups is known to have a solvable word problem and it includes many important families which arise naturally in geometry and topology (e.g. hyperbolic and Euclidean groups). The work of these authors represents one of the first major practical successes of the KB-procedure in the study of fp-groups. Recently, Holt & Rees extended these techniques so as to produce a method for determining isomorphism of pairs of finitely presented groups. While the method may have a rather low success rate when applied to an arbitrary pair of fp-groups, when applied to a special class of groups such as automatic groups, it can be much more successful. For example, the authors report that it was able to settle the isomorphism question for all but two pairs in a collection of about 30 pairs of link groups.

In the paper titled "The Knuth-Bendix procedure for strings as a substitute for coset enumeration", Sims employs the KB-procedure to deduce non-obvious relations in two fp-groups. In each case, the discovery of these relations could not have been accomplished using the current generation of TC-procedures. This is one of the first reported instances where the KB-procedure outperforms the TC-procedure and is significant for that reason.

Over the past two decades, a considerable number of useful permutation group algorithms have been discovered. Let G be a permutation group acting faithfully on the

finite set Ω and suppose G is given in terms of a small set of generating permutations X. The following are representative of the type of information sought by permutation group theorists:

Does the group G act transitively (primitively, regularly) on the set Ω?
What is the order of the group G?
Find generators for the stabilizer of a sequence (set) of elements of Ω.
Determine the various series of characteristic subgroups of G: derived series, lower central series, upper central series.
Find generators for the Sylow p-subgroup of G, where p is a prime dividing the order of G.
Compute the centralizer (normalizer, normal closure) of a subgroup H of G.
Determine representatives for the conjugacy classes of elements of G.
Compute a composition series for G and determine the isomorphism type of each composition factor.

All but the first of these questions involve quantifying over the *set* of elements of G. The fundamental notion of a base and strong generating set (BSGS) introduced by Sims in 1970, provides a very compact representation of this set. A *base* for G is a sequence $B = \beta_1, \ldots, \beta_k$ of distinct elements of Ω such that the identity is the only element of G that fixes B pointwise. Thus, B defines a sequence of subgroups

$$G = G^{(1)} > \cdots > G^{(k+1)} = \langle 1 \rangle$$

where $G^{(i)} = G_{\beta_1,\ldots,\beta_{i-1}}$. A *strong generating set* S for G, relative to the base B, is a generating set for G which contains generators for each subgroup in the chain. Given B and S, it is a straightforward matter to compute the orbit $\Delta_i = \beta_i^{G^{(i)}}$ and a transversal $U^{(i)}$ for $G^{(i+1)}$ in $G^{(i)}$ for $i = 1, \ldots, k$. Knowledge of Δ_i, $U^{(i)}$ for $i = 1, \ldots, k$ immediately gives us the order of G, a membership test for G, and the possibility of listing the elements of G without repetition.

The design of fast methods for constructing a BSGS for G is one of the central problems in computational permutation group theory. In 1967, Sims developed a BSGS algorithm that was based on a lemma of Schreier. Given a subgroup H of G, this lemma gives a generating set for H in terms of a transversal for H in G and generators of G. The running time of this algorithm was bounded by $O(n^6)$, where n is the degree of G. In 1980, Jerrum described a variant of the original Sims algorithm with running time $O(n^5)$. The Sims algorithm works well for degrees less than 100 but becomes impractical as the degree increases beyond a few hundred. A different approach involves first constructing a "probable" BSGS for G, and then applying an algorithm which either *verifies* that the BSGS is correct, or establishes that it is incomplete. A "probable" BSGS may be constructed very quickly. The first verification algorithm was developed by Leon in 1976 and involves using the TC-procedure to construct presentations for the successive terms of the stabilizer chain, starting at the bottom. This method, sometimes referred to as the *Schreier-Todd-Coxeter algorithm*, made it practical to construct BSGSs for groups having degree in the low thousands. In 1986, Brownie, Cannon & Sims implemented a new verification algorithm which has been successfully applied to groups of degree up to 500 000.

Cooperman & Finkelstein in their paper "A strong generating test and short presentations for permutation groups", describe an algorithm which verifies strong generation in $O(n^4)$ time. More experimental work needs to be done in order to establish whether or

not the Cooperman-Finkelstein algorithm is a practical competitor to the Brownie-Cannon-Sims algorithm.

The availability of the BSGS representation of a permutation group provides the appropriate foundation for the design of efficient backtrack searches for subgroups of G whose elements satisfy some elementary property. The BSGS backtrack search of a permutation group was introduced by Sims in 1970, when he described backtrack algorithms for computing centralizers and intersections of subgroups. Over the next decade, Butler *et al.* developed backtrack searches for set stabilizers, normalizers, Sylow p-subgroups and for testing conjugacy of elements and subgroups. These backtracks are currently the backbone of the permutation group machinery in the Computer Algebra system CAYLEY. In his paper "The computation of normalizers in permutation groups", Holt presents a backtrack algorithm which employs many additional tests to prune the backtrack search tree. The performance of his algorithm is superior in many cases to the Butler algorithm as implemented in CAYLEY.

Leon's paper, "Permutation group algorithms based on partitions, I: theory and algorithms", represents a major step in the evolution of backtrack algorithms for permutation groups. The efficiency of a backtrack search is heavily dependent upon the information available to prune the search tree. Using the idea of partition refinement, first developed by Brendan McKay as part of his highly successful graph automorphism algorithm, Leon is able to construct very powerful tests. An early implementation of a set stabilizer algorithm based on these ideas demonstrates performance that is vastly superior to the "first generation" set stabilizer algorithm. As a result of this work we can expect a new generation of backtrack algorithms, exhibiting superior performance, to emerge in the near future.

Let p be a prime dividing $|G|$, and let P denote the Sylow p-subgroup of G. Traditional approaches to computing P have involved performing a series of cyclic extensions commencing with an element of p-power order. In "Computing Sylow subgroups of permutation groups using homomorphic images of centralizers", Butler & Cannon present a recursive method based on reduction of the degree. The reduction is based on the observation that if z is a p-central element having order p, then the kernel of the action of the centralizer of z on the cycles of z is a p-group.

Once one has access to constructions such as centralizer, normal closure and Sylow subgroup, one can contemplate computing a description of the abstract structure of G. Such a goal became practical with the completion of the classification of the finite simple groups in 1982. In 1984, at the Groups_St Andrews meeting, P. Neumann described a practical algorithm, based on the O'Nan-Scott theorem, for determining a BSGS for each composition factor of G. The general strategy involved reducing to a primitive group T, locating the socle of T, and then splitting the socle into its simple direct factors. In 1987, Luks published a polynomial time algorithm for this problem which appears to be of theoretical interest only. Neumann's algorithm made some use of the fact that practical computation with such an algorithm will be restricted to groups having degree at most of a few million. In the paper, "Finding composition factors of permutation groups of degree $n \leq 10^6$", Kantor pushes this approach somewhat further and shows that it is possible to name the composition factors for groups of degree not exceeding 10^6 at the cost of computing only a BSGS for G and the derived subgroup of G (except in some rather rare cases). Thus, Kantor avoids having to construct and split the socle of T.

The paper "Fast recognition of doubly transitive groups" by Cameron & Cannon presents an algorithm for identifying a doubly transitive group. By carefully analysing

the lengths of the orbits of a two-point stabilizer (three-point stabilizer in a triply transitive group), the algorithm avoids having to compute the derived subgroup of G, except in the case of some relatively small groups (e.g. one-dimensional affine groups).

Finally, Leedham-Green *et al.* in their paper, "Computing with group homomorphisms", describe a simple and elegant method of determining whether a mapping between two finite groups is a homomorphism, and if it is, of computing its kernel.

I would like to express my gratitude to the authors and referees for the great deal of hard work involved in the production of this collection of papers. I would also like to thank Derek Holt and Cheryl Praeger who acted as editors of those papers for which I was coauthor. Finally, it gives me great pleasure to thank Bruno Buchberger for encouraging the publication of these two special issues.

John Cannon